# Outline for May 5, 2005

**Reading**: §12.3–12.6, §22.2, §15

## Discussion

It has often been said tha the only way to decipher a message that has been enciphered using RSA is to factor the modulus $n$ used by the cipher. If you were told that an enciphered message was on a computer that you controlled, and that the message was enciphered using RSA with an $n$ of 1024 bits (about 309 decimal digits), how would you find the encrypter's private key?

## Outline

1. Challenge-response systems
   a. Computer issues challenge, user presents response to verify secret information known/item possessed
   b. Example operations: $f(x) = x+1$, random, string (for users without computers), time of day, computer sends $E(x)$, you answer $E(D(E(x))+1)$
   c. Note: password never sent on wire or network
   d. Attack: man-in-the-middle
   e. Defense: mutual authentication
2. Biometrics
   a. Depend on physical characteristics
   b. Examples: pattern of typing (remarkably effective), retinal scans, *etc*.
3. Location
   a. Bind user to some location detection device (human, GPS)
   b. Authenticate by location of the device
4. Combinations: PAM
5. Access Control Lists
   a. UNIX method
   b. ACLs: describe, revocation issue
6. Capabilities
   a. Capability-based addressing: show picture of accessing object
   b. Show process limiting access by not inheriting all parent's capabilities
   c. Revocation: use of a global descriptor table
7. Privilege in Languages
   a. Nesting program units
   b. Temporary upgrading of privileges
8. Lock and Key
   a. Associate with each object a lock; associate with each process that has access to object a key (it's a cross between ACLs and C-Lists)
   b. Example: use crypto (Gifford). $X$ object enciphered with key $K$. Associate an opener $R$ with $X$. Then:
      OR-Access: $K$ can be recovered with any $D_i$ in a list of $n$ deciphering transformations, so
      $R = (E_1(K), E_2(K), ..., E_n(K))$ and any process with access to any of the $D_i$'s can access the file
      AND-Access: need all $n$ deciphering functions to get $K$: $R = E_1(E_2(...E_n(K)...))$
   c. Types and locks
9. MULTICS ring mechanism
   a. MULTICS rings: used for both data and procedures; rights are REWA
   b. $(b_1, b_2)$ access bracket - can access freely; $(b_3, b_4)$ call bracket - can call segment through gate; so if $a$'s access bracket is (32,35) and its call bracket is (36,39), then *assuming permission mode (REWA) allows*

*access*, a procedure in:
rings 0-31: can access *a*, but ring-crossing fault occurs
rings 32-35: can access *a*, no ring-crossing fault
rings 36-39: can access *a*, provided a valid gate is used as an entry point
rings 40-63: cannot access *a*

c. If the procedure is accessing a data segment *d*, no call bracket allowed; given the above, *assuming permission mode (REWA) allows access*, a procedure in:
rings 0-32: can access *d*
rings 33-35: can access *d*, but cannot write to it (W or A)
rings 36-63: cannot access *d*